System Implementation[1]

Web-based Submission of the Discharge Monitoring Report[2]

EPA Contract #68-W5-0030[3]

Delivery Order #0004

Revised August 30, 1999

[1] Deliverable 3.2, Information Dynamics, Inc.

[2] A field test in the State of New York of the digital signing and submission of the Discharge Monitoring Report using an Adobe Acrobat Exchange plug-in to a Web browser as the electronic form environment which is connected interactively across the Internet to a receiving Web site. Cryptographic and handwritten biometric digital signatures are evaluated in this pilot.

[3] Submission of Environmental Data Under the Taiwan-USEPA Technical Cooperation Agreement

1 Scope

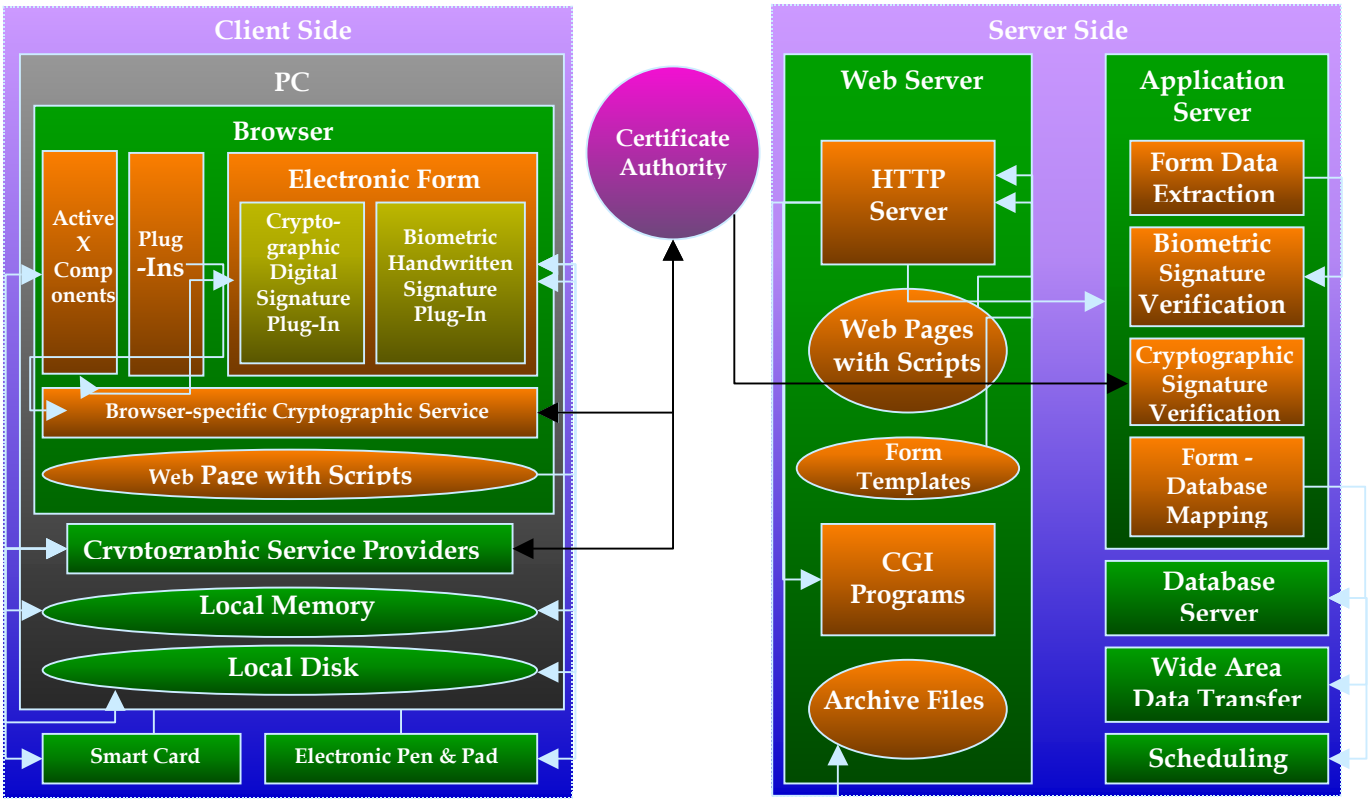
This System Implementation Document describes how the functionality of commercial off-the-shelf products was implemented to create an integrated prototype system for the pilot test of the Web-based submission of the New York State Discharge Monitoring Report (DMR) conducted in the State of New York June – November, 1999.

Functional descriptions of each system component, and selection of commercial off-the-shelf (COTS) products to provide the required component functionality are found in the Requirements Document. The way in which the pilot participants use the system implemented for the DMR pilot is discussed in the Design Document.

2 System Integration of Component Functions

The following diagram illustrates how component functions are integrated to form the prototype system for signing and submitting electronic DMRs. In the highest level view, the system implementation consists of:

- hardware and software components installed on the computers used by the submitters (the client side),
- the certificate authority's servers and functions,
- software components installed at the receiving Web site (the server side).



At the next level of detail, the client side can be subdivided into the following components:

- the personal computer (PC),
- peripheral hardware components (e.g., a smart card reader or an electronic pen & pad).

The personal computer, in turn, can be considered a container for the following functions:

- Web browser,
- cryptographic service providers,
- local memory,
- local disk.

The Web browser, in turn, can be treated conceptually as encompassing the following functions, in the sense that the Web browser provides a software context within which the following components can run:

- electronic form,
- ActiveX components,
- plug-ins,
- browser-specific cryptographic service,
- Web pages with scripts.

In the most detailed view of the integrated system, the electronic form can be treated as a container which provides a context for the following components:

- cryptographic digital signature plug-in,
- biometric handwritten signature plug-in.

A similar "Chinese box" hierarchical relationship of conceptual containers of component functionality can be applied to the server side. At the highest level, the server side can be subdivided into the following components:

- Web server,
- application server,
- database server,
- wide-area data transfer,
- scheduling.

The Web server, in turn, can be thought of as encompassing the following components:

- HyperText Transfer Protocol (HTTP) server,
- Web pages with scripts,
- form templates,
- Common Gateway Interface (CGI) programs,
- archive files.

The application server can be considered as containing the following functions needed to implement the DMR pilot:

- form data extraction,
- biometric signature verification,
- cryptographic signature verification,

- form-database mapping.

The above component functions, conceptualized at different levels of detail, are integrated within the prototype system to fill various roles in the DMR pilot. For example, the client side functions run on the pilot participant's computers and are used by the pilot participants to display, edit, sign and submit DMR forms. The pilot participants use client-side functions to enroll their identity information with the certificate authority and complete the registration of their certificates. The New York State Department of Environmental Conservation (NYS DEC) uses the client-side functions to access the certificate authority's server to fulfill their role as Local Registration Authority for the pilot. NYS DEC also uses the client-side functions to display submitted DMRs.

Server-side functions, implemented at the DMR pilot's receiving Web site, are used to validate login IDs and passwords, configure Web pages based on the pilot participant's login ID and menu selections, select the DMR form template and send the appropriate data across the Internet to the pilot participant's computer to pre-populate the DMR form with the correct default values. Server-side functions also are used to receive submitted data, authenticate digital signatures, store the submitted data in the correct tables and fields of a database, and transmit submitted data to NYS DEC according to an established batch data transfer schedule.

At a more detailed level, the integration of component functionality to implement outcomes specific to the DMR pilot is discussed in the following sections.

2.1 Client-side Electronic Form

The client-side electronic form functionality is implemented with Adobe Acrobat Exchange Version 3.01 as a plug-in to the submitter's Web browser. For the purposes of the DMR pilot, the Web browser is implemented using Netscape Navigator 4.51. The electronic form allows the pilot participants to view, edit, save and submit DMR forms. The electronic form exchanges data with the DMR pilot's receiving Web site through Adobe's Forms Data Format (FDF) standard which in turn is packaged within the HyperText Transfer Protocol (HTTP) data exchange between the pilot participant's Web browser and the receiving Web site.

2.2 Client-side Digital Signature

The client-side digital signature functionality will be implemented differently for Phase 1 and Phase 2 of the DMR pilot. In Phase 1, a cryptographic digital

signature will be used, and in Phase 2 a biometric handwritten digital signature will be used.

2.2.1 Cryptographic Digital Signature

The cryptographic digital signature function is implemented on the client-side by means of a plug-in to the electronic form. In the DMR pilot, the cryptographic digital signature plug-in to the Adobe Acrobat Exchange form is supplied as part of E-Lock Technologies' Assured Transactions solution. The cryptographic digital signature plug-in depends upon application programming interfaces (APIs) provided by the Adobe Acrobat Exchange form to report the content of the form to the cryptographic digital signature plug-in, since the content of the form (template plus data) is one of two inputs to the digital signature algorithm.

The other input required by the digital signature algorithm, a private cryptographic key, is provided to the cryptographic digital signature plug-in by the Microsoft Cryptographic Application Programming Interface (CryptoAPI) which is available in the Windows 95, 98 and NT operating systems. [Internet Explorer 4.01 or greater must be installed to enable the operating system's cryptographic services in Windows 95 and NT, even if Internet Explorer is not used for its Web browser function.] CryptoAPI allows software applications running in a 32-bit Windows environment to access installed cryptographic service provider modules in a uniform manner. In the DMR pilot, the cryptographic service provider is implemented by installing GemSAFE software provided by Gemplus, which accesses the Gemplus smart card to generate the private cryptographic key and provide this key to the E-Lock cryptographic digital signature plug-in via CryptoAPI.

After the cryptographic digital signature plug-in has calculated the digital signature hash based on the content of the form and the private cryptographic key, the digital signature hash value is stored in a hidden field of the electronic form and is transmitted to the receiving Web site as part of the FDF data stream.

A more detailed explanation of the implementation of the cryptographic digital signature plug-in within the context of the DMR pilot can be found in Appendix A, Sections 3.1.1 and 3.4.

2.2.2 Biometric Handwritten Digital Signature

The biometric handwritten digital signature function is implemented as a plug-in to the electronic form. In the DMR pilot, this function is provided by PenOp. When the biometric handwritten digital signature plug-in is activated at the time the signer clicks an icon in the electronic form, the plug-in displays a prompt to the signer and receives signature dynamics data from a CalComp UltraSlate

graphics tablet and pen connected to the signer's computer through a serial data interface.

The biometric handwritten digital signature plug-in binds the signature dynamics data (e.g., acceleration, wobble, timing, etc.) to the contents of the form (supplied to the plug-in via Adobe APIs) using a cryptographic algorithm (MD5), and stores the resulting digital signature in hidden fields of the electronic form to be transmitted to the receiving Web site in the FDF data stream.

2.3 Server-side Application Server

The application server is a server-side software component of the receiving Web site in the DMR pilot. The application server receives for special processing data that the HyperText Transfer Protocol (HTTP) Server has in turn received across the Internet from the client-side Web browser on the pilot participant's computer. In this sense, the application server can extend the capability of the HTTP server beyond the publishing of static Web pages. In the DMR pilot, the application server assembles database-driven Web pages to create selection menus for the pilot participants.

The application server also manages the mapping between fields in the electronic form and fields in the receiving Web site's database, so that data can be exchanged between the receiving Web site and the electronic form on the client computer. Through programming logic, the application server manages the flow of actions performed at the receiving Web site, including the launching of external applications (such as digital signature verification) and directs the workflow that the pilot participant experiences when connecting to the receiving Web site.

In the DMR pilot, the application server function is provided by HAHTsite Application Server Version 3.1 from Haht Software, Inc. The HAHTsite application server accesses the Web site's database server (Microsoft SQL Server Version 6.5) via SQL commands presented to the database server using the Open Database Connectivity (ODBC) standard.

2.4 Server-side Digital Signature Verification

Digital signatures are verified at the receiving Web site by means of external server-side applications called by the application server. The verification of digital signatures requires that the signature verification application have access to the same DMR form content which was provided to the digital signature algorithm on the signer's computer. This means that this form content must be reproduced at the receiving Web site. The application server assists in the server-side reconstruction of the DMR content through programming which

identifies which DMR form template was provided to the signer and which DMR data set was received from the signer. The application server then passes this form content information to the signature verification application.

In the case of verifying biometric handwritten digital signatures, it is also necessary to load the DMR form template and data into the electronic form application (Adobe Acrobat Exchange) running at the receiving Web site and activate a plug-in (provided by PenOp) to the electronic form application on the server side to verify the signature.

The process of verifying cryptographic signatures using a software component provided by E-Lock Technologies is described in Appendix A, Sections 3.1.2, 3.4 and 3.4.4.

2.5 Server-side Batch Data Transfer

The function which exchanges data between the receiving Web site established for the DMR pilot and the New York State Department of Environmental Conservation is fulfilled for the purposes of the pilot by structured text files encapsulated in E-mail attachments. A Cold Fusion application mediates between the Web site's database server and E-mail attachments through programming which issues SQL commands to the database server and constructs attachments to Simple Mail Transfer Protocol (SMTP) E-mail messages which are sent via a Linux Sendmail relay host on a predefined schedule.

The rudimentary batch data transfer mechanism established for the DMR pilot could be replaced by a more full-featured Electronic Data Interchange (EDI) mechanism if desired.

2.6 Certificate Authority

The certificate authority service for the DMR pilot is provided by E-Lock Technologies, Inc., which maintains a certificate authority server at E-Lock's corporate location. E-Lock's certificate authority server is accessible via the Internet by the pilot participants for the purpose of enrolling their identity information and registering their certificates, and also by the New York State Department of Environmental Conservation for the purpose of approving and managing certificates. The role of E-Lock's certificate authority service in the context of the other functions of the DMR pilot is depicted in the diagram found in Appendix A, Section 3.1.

The administrative console for use by the New York State Department of Environmental Conservation (when acting as the Local Registration Authority for the DMR pilot) is implemented by using the Web browser, Microsoft Internet Explorer 4.01, to access interactive Web pages maintained by the certificate

authority. Access to the certificate authority server for the purpose of Local Registration Authority (LRA) functions is controlled by a smart card in a smart card reader attached to the LRA administrator's computer. The smart card is used for client authentication of the LRA administrator's Web browser to the certificate authority's Web server over a Secure Sockets Layer (SSL) connection.

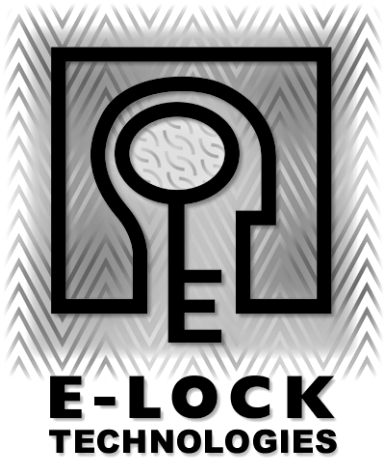
The process by which the pilot participants enroll their identify information (the first step in the certificate request procedure) is implemented using a standard HTML Web form maintained by the certificate authority to receive input from the pilot participant's Web browser.

The certificate process is implemented using two alternative methods:

- Client-side software is installed on the pilot participant's computer which accesses the certificate authority's server across the Internet to join the participant's identity information previously received by the certificate authority with a public key generated by the participant's smart card when a one-time access code is entered by the participant to create a completed certificate which is then transmitted to the certificate authority's server.
- ActiveX software components automatically downloaded to the participant's Internet Explorer 4.01 Web browser duplicate the functions of the client-software as described above. Using the Web browser registration method, all Internet communications are controlled by the Web browser's configuration settings, which allow the network data transmissions to pass through most firewalls.

Appendix A

3 System Implementation of E-Lock Components



EPA DMR Project

Design and Architecture

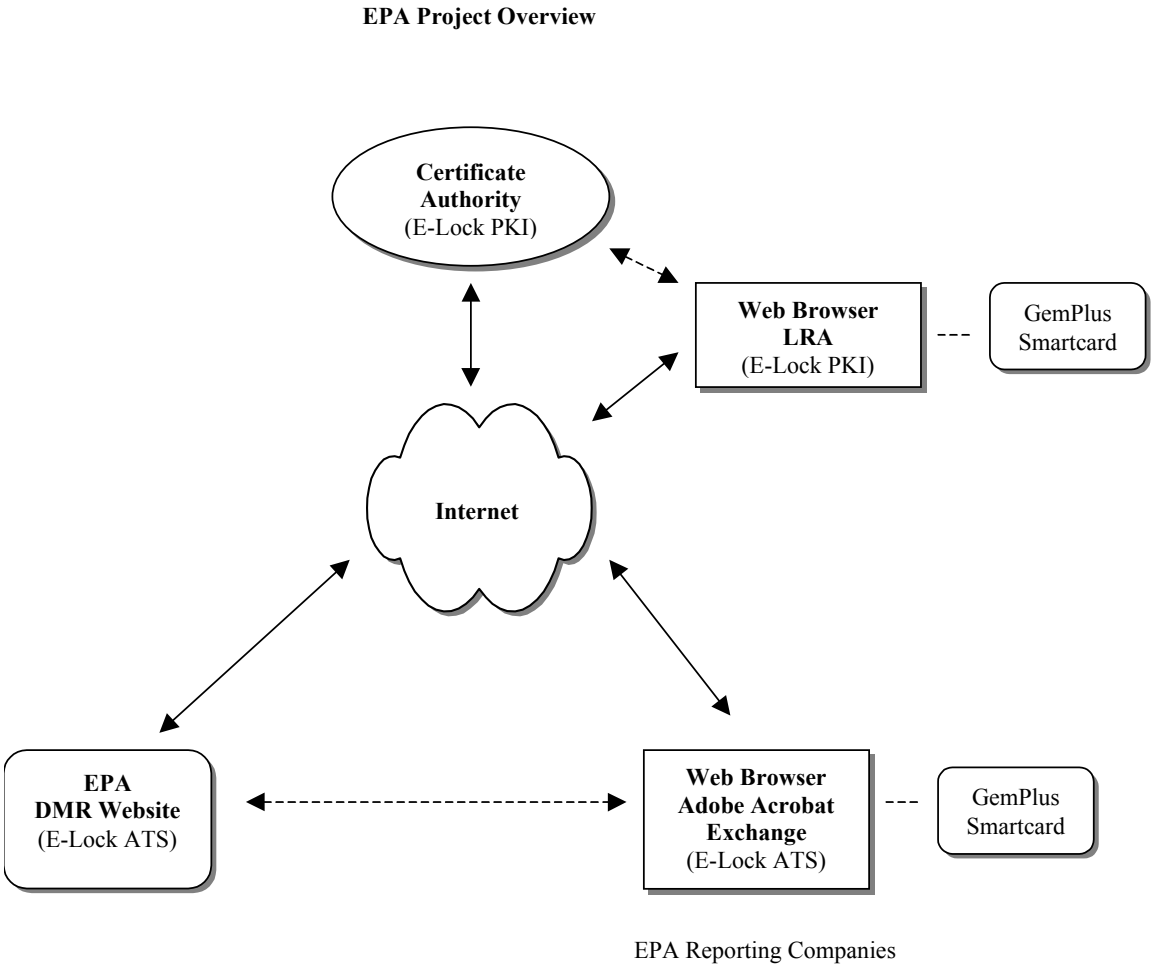
Revised: August 27, 1999

Adobe Acrobat Form Signing

Adobe Acrobat Form Signature Verification (Server Side)

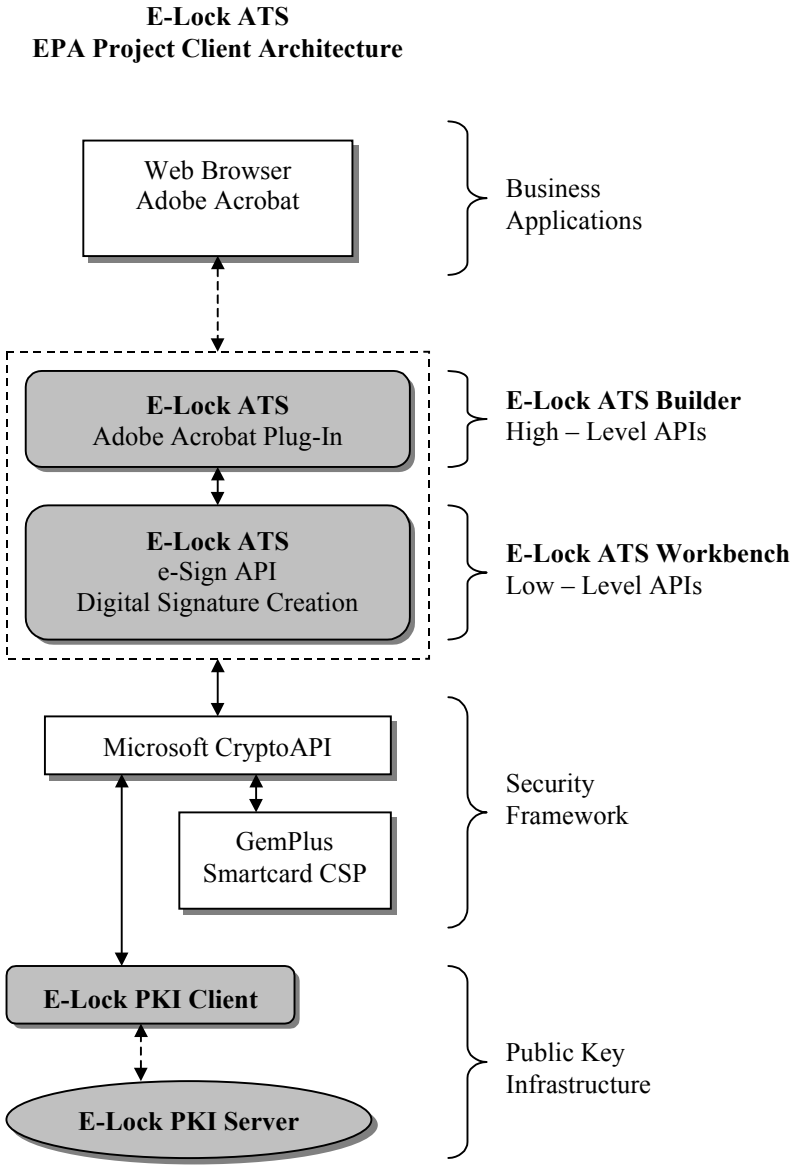
3.1 Introduction

This document discusses the design and architecture of the E-Lock Technologies PKI and Adobe Acrobat form signing products and services used for the EPA DMR Pilot. An overview of how E-Lock components are used in the EPA DMR pilot is shown below:



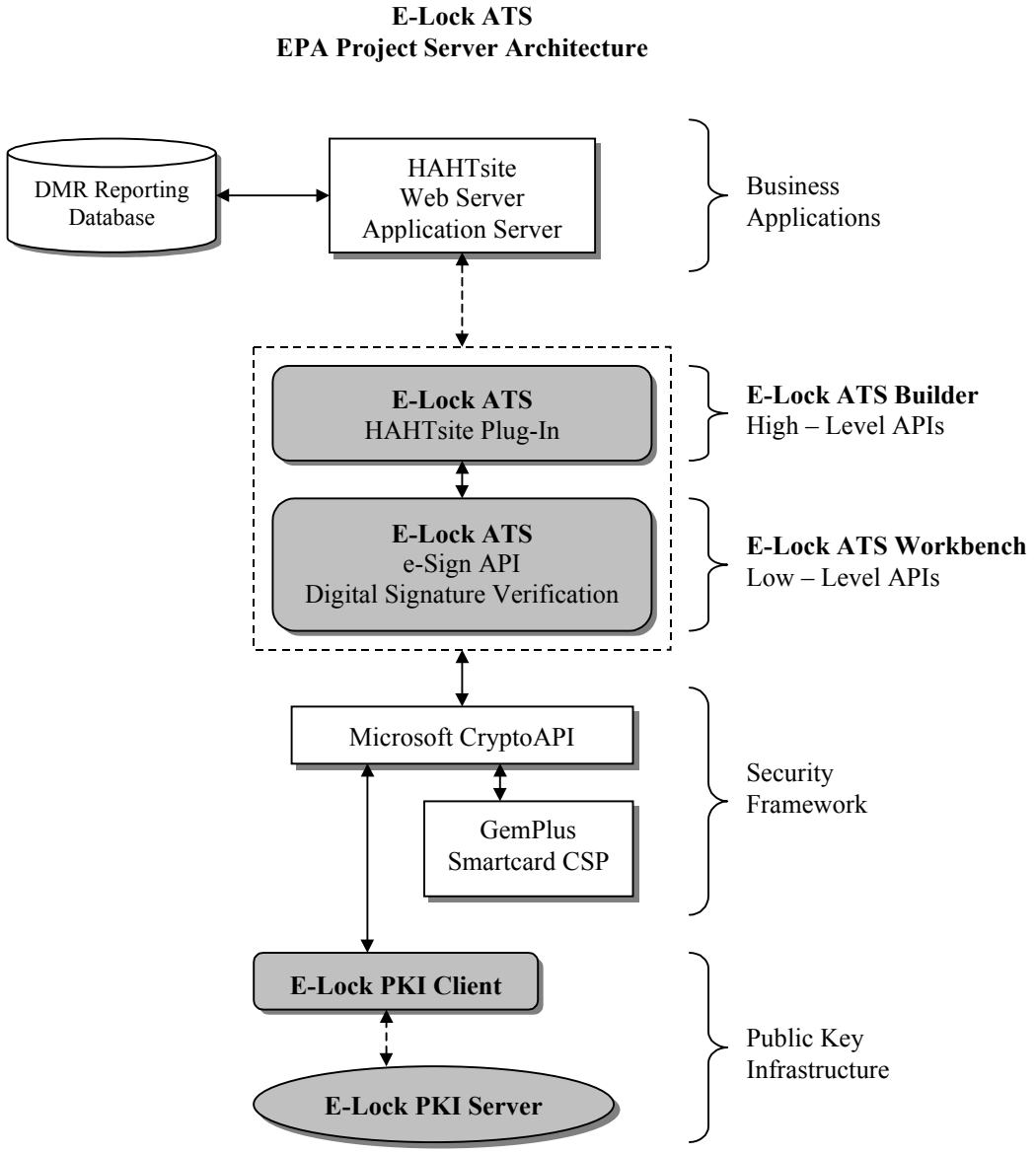
3.1.1 EPA DMR Pilot Client Architecture

The following diagram shows the architecture of E-Lock's Assured Transactions products as they are implemented on the client computer used in the EPA DMR pilot.



3.1.2 EPA DMR Pilot Server Architecture

The following diagram shows the architecture of E-Lock's Assured Transactions products as they are implemented on the receiving Web site server used in the EPA DMR pilot.



3.2 Abstract

E-Lock Technologies has integrated its E-Lock ATS digital signature technology into Adobe Acrobat forms, providing data-origin authentication and data-integrity services to the user of Adobe Acrobat PDF forms. The purpose of this document is to explain the E-Lock Adobe Acrobat Form Signing solution, and to give a comprehensive outline, from a user's* point of view, for the same. Note: *"USER" in this context is the one who will author the Adobe Acrobat Forms and make them available on the Web to be accessed, filled and submitted.

Before going into the details, it is necessary to get acquainted with a few facts about the Adobe Acrobat architecture first.

3.3 A Brief Note about Adobe Acrobat Architecture

Adobe has a well-known proprietary document format called the Portable Document Format (PDF). Adobe uses the same format for its web forms. A PDF form can be viewed in the browser by having a plug-in for Netscape Communicator and an ActiveX control for Microsoft Internet Explorer. The plug-in and ActiveX control have to be complemented by proprietary extensions to the Adobe Acrobat Exchange. These proprietary components have an extension ".API" and are loaded by the Adobe Acrobat Exchange plug-in or the ActiveX controls.

While creating a PDF form, Adobe Acrobat allows third party vendors to add their own objects, referred to as an "annotation" to the form. These annotations are implemented using the API extension DLLs.

Using the annotation concept, E-Lock Technologies ATS provides data integrity and data-origin authentication services to the Adobe Acrobat forms.

Each Adobe Acrobat web form has a E-Lock Digital Signature annotation, which the end-user can invoke to sign and verify the form at any instance before the form is submitted. Later the form can be submitted along with the digital signature, to be verified at the server side and the form data and signature can be stored on the server for the purpose of non-repudiation.

The design of the Adobe Acrobat plug-in (extension) centers around a few Adobe technologies. Of most importance to the transport layer is the Acrobat Form plug-in. Adobe provides an Acrobat Forms Author plug-in which allows fields to be added to a PDF file. With the help of this forms Author plug-in, the fieldnames and their values can be enumerated at any time, and their contents can be saved with the PDF file. The fields can also be sent in a proprietary Field Data Format (FDF) to a server, which can process the fields and even remotely fill them in with new data.

3.4 The E-Lock Solution for Digitally Signed Adobe Acrobat Web Forms

This section will discuss how the Adobe Acrobat Web form signing is implemented by E-Lock ATS. Figure 1 depicts what happens when an Adobe Acrobat form is digitally signed and verified.

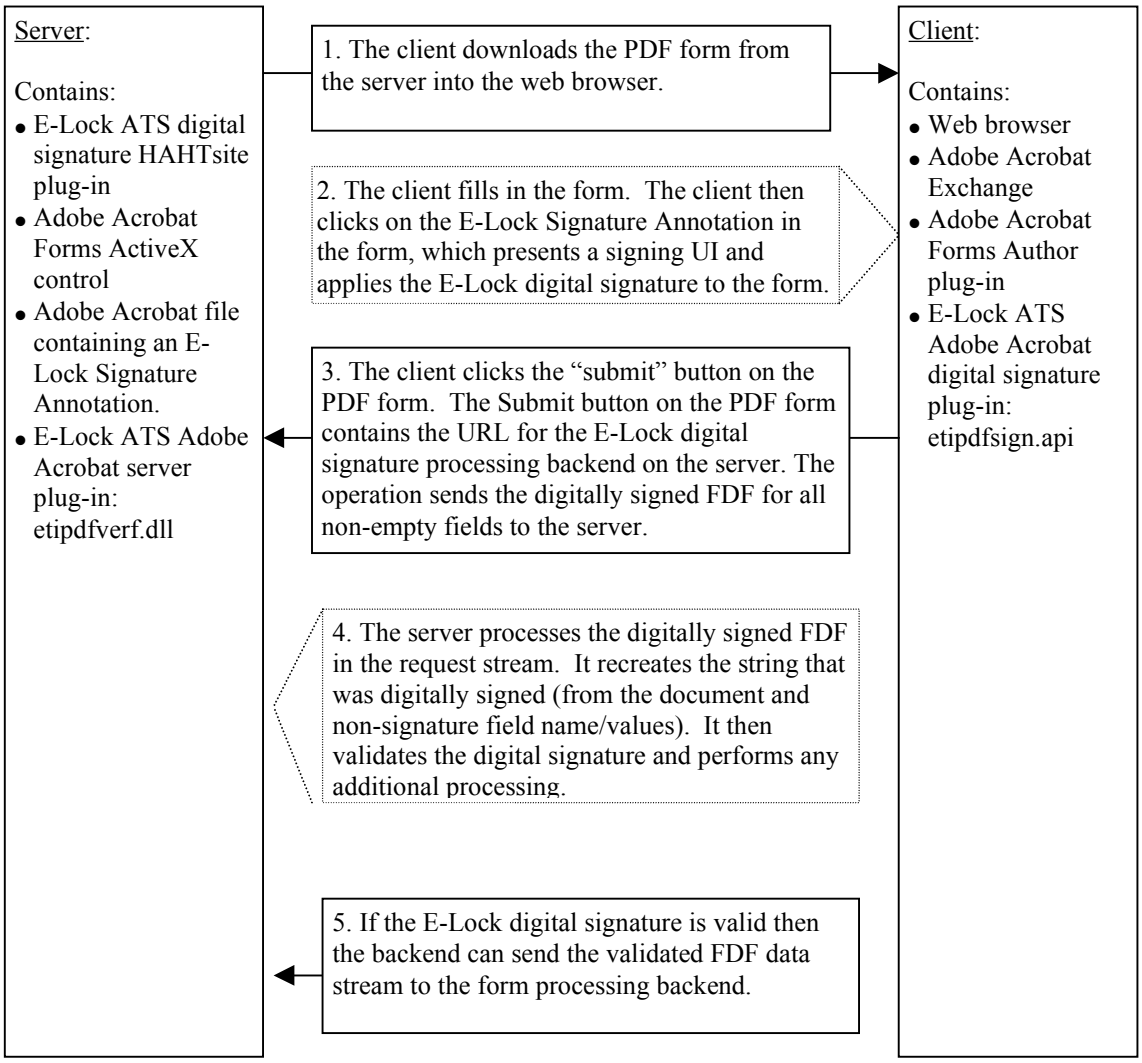


Figure 1 **E-Lock ATS Form Signing Transport Architecture**

3.4.1 E-Lock PDF Form Digital Signature Details

Using the E-Lock Form Digital Signature plug-in users can click on the signature annotation displayed in the PDF Form (as displayed in a browser) to digitally sign the form. The E-Lock Adobe Acrobat plug-in uses the Adobe FDF toolkit to enumerate all the non-null FieldName – FieldValue pairs in the Acrobat form and concatenates them to form a string. It then BASE64 encodes the original PDF form data and appends it to the FDF data to this string. The E-Lock plug-in is

then used to digitally sign this complete concatenated string and saves the result in a hidden field designated to store the digital signature. When the user presses the submit button on the Acrobat form which has an associated action of "Submit form data to Server", the Acrobat Form plug-in creates an FDF stream of all the FieldName-Fieldvalue pairs along with the E-Lock hidden digital signature field and sends this FDF stream to the server. At the Server end, the FDF stream is passed to the E-Lock HAHTsite backend which is using the ETIPDFVERF.DLL. Using the FDF stream and the PDF form file residing at the Server the backend regenerates the data which was digitally signed at the client and verifies this against the digital signature stored in the hidden field which was transported from the client.

3.4.2 Adobe Acrobat Form Authoring

For creating Adobe Acrobat PDF Forms one needs to have the Acrobat Form Author Plug-in installed with Adobe Acrobat Exchange. The Form Author Plug-in can be freely downloaded from the Adobe web site. The first step after creation of all the form fields is to add the E-Lock Digital Signature Annotation to the desired place on the Adobe Acrobat Form. The Author / publisher of the Adobe Acrobat Form will have to copy the e-Lock digital signature plug-in (ETIPDFSIGN.API) into Adobe Acrobat exchange plug-ins directory. This .API extension implements a digital signing annotation which shows up as an E-Lock annotation bitmap embedded in the digital signature enabled PDF web form.

1. **Adding the E-lock Signature Annotation**

   When the user opens the Adobe Acrobat Exchange for Authoring the form, the toolbar will show the buttons to add the E-Lock Digital Signature annotation to the form.

2. **Adding a hidden field ZELockSignature1 to the form**

   The form-author would create a hidden field with the name *ZELockSignature1* and should place it exactly below the E-Lock Signature annotation icon. The width and height of this hidden field should be more than the E-Lock Signature annotation. This hidden field is required to transport the form signature to the server-end. One has to do this manually (at the time of placing the E-Lock annotation) because Adobe Acrobat does not support dynamic creation of form fields.

3. **Setting the action for Submit button**

   The form-author would specify the URL that will invoke the E-Lock Adobe Acrobat Form Verification control to verify the signature.

3.4.3 Adobe Acrobat Form Signing

The user can access the PDF sign-enabled web form using a browser, fill in the details of the form and then digitally sign them. To digitally sign the form, the user has to be in Adobe Acrobat's normal input mode (the hand cursor). When the cursor goes over the digital signature, it changes. Clicking on the object effectively "digitally signs" it. The ETIPDFSIGN.API module accesses the PKI information present on the client machine using E-Lock ATS controls and provides an interface to select the following:

- The Cryptographic Service Provider(CSP) and CSP Type
- The Private Key
- The Hash Algorithm

The user can select form the above details. The ETIPDFSIGN.API now digitally signs the form contents and stores the digital signature data in a hidden field *ZELockSignature1*. The digital signature is then verified at the client, which changes the annotation bitmap depending on the verification results. If the form has already been digitally signed, moving the cursor over the E-Lock digital signature annotation changes the cursor. Clicking on the E-Lock annotation will cause a re-verification. So, if the Adobe Acrobat form is digitally signed and a field is subsequently changed, clicking the E-Lock digital signature annotation will verify the digital signature using the new data in the form and a dialog will be displayed which indicates "contents were altered after signing". This is an indication that the user needs to re-sign the form.

3.4.4 Adobe Acrobat Form Signature Verification (Server side)

The PDF form contains a URL to which to submit the FDF data. This URL invokes the E-Lock HAHTsite backend. This subsequently calls the E-Lock Adobe Acrobat digital signature verification component (ETIPDFVERF.DLL). The E-Lock server components process the FDF data, and verify the digital signature. They get information about the verification status, the name of the original PDF document, and can enumerate the sent/signed field name/value pairs.